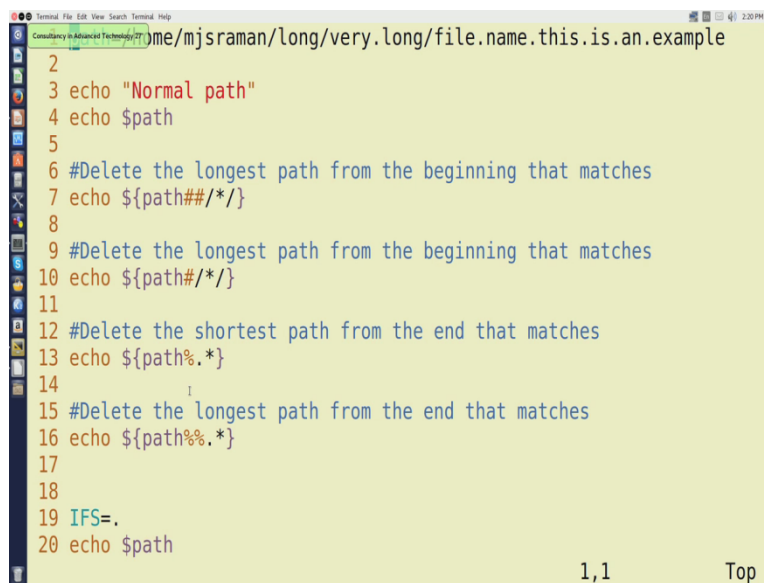


Information Security 3
Sri M J Shankar Raman,
Consultant Department of Computer Science and Engineering,
Indian Institute of Technology Madras
Module 37
Shell pattern matching

Hi there, welcome to this session on shell scripting. In the last session we actually saw how to use the for loop, now before we go to the next example.

(Refer Slide Time: 00:37)



```
me/mjsraman/long/very.long/file.name.this.is.an.example
2
3 echo "Normal path"
4 echo $path
5
6 #Delete the longest path from the beginning that matches
7 echo ${path##*/}
8
9 #Delete the longest path from the beginning that matches
10 echo ${path#*/}
11
12 #Delete the shortest path from the end that matches
13 echo ${path%.*}
14
15 #Delete the longest path from the end that matches
16 echo ${path%%.*}
17
18
19 IFS=.
20 echo $path
```

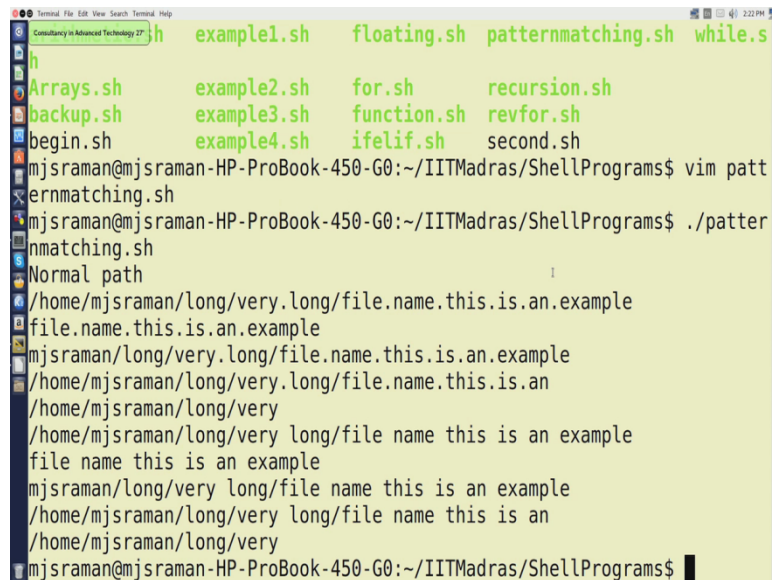
1,1 Top

We will give you very brief introduction to pattern matching in shell scripts, before we take a look at the next example. We believe that this will be of use to you; this is just an introduction to pattern matching. You would've seen other types of pattern matching commands like star are being used, but this is specific for shell script and this relates on how to manipulate variables using pattern matching using the dollar dollar symbol, ok?

So let us try for example. Let's take for example a path name ok, so a path name usually look something like this say for example slash home slash mjsranan slash long slash very dot long slash file dot name dot this,this an example bla bla bla, so this this is something like this. Now, this is a single string ok, now let us see how we can split this string ok and then match patterns based on these examples, see otherwise I mean it's very difficult to understand how a pattern

matching actually works, so let us check this so, when you execute the shell script echo dollar part, we will see that this whole string will be displayed ok. So lets start of that and then we will try to understand how these pattern matching works, ok this is essential to understand the next example therefore lets try to execute this program and then see what happens, let us first execute this program, like so.

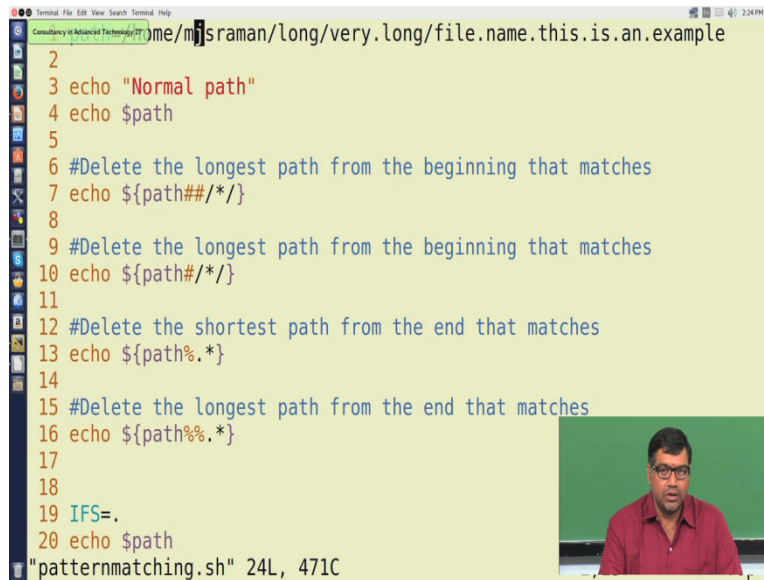
(Refer Slide Time: 02:19)



```
h example1.sh floating.sh patternmatching.sh while.s
h
Arrays.sh example2.sh for.sh recursion.sh
backup.sh example3.sh function.sh revfor.sh
begin.sh example4.sh ifelif.sh second.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patternmatching.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./patternmatching.sh
Normal path
/home/mjsraman/long/very.long/file.name.this.is.an.example
file.name.this.is.an.example
mjsraman/long/very.long/file.name.this.is.an.example
/home/mjsraman/long/very
/home/mjsraman/long/very
/home/mjsraman/long/very long/file name this is an example
file name this is an example
mjsraman/long/very long/file name this is an example
/home/mjsraman/long/very long/file name this is an
/home/mjsraman/long/very
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$
```

So let's start so this example so as I told you once we executes the program it says normal path ok and then normal path you have printed the path variable which is nothing but m home mjsraman long slash very dot long and so on. So what is given in this the the third line, the next line after the normal path, Now let us look at the example ok and if you remember just remember this that we are using file dot name dot this is what getting printed after the next command. So let us see how this so, which essentially means they are removing the slash home slash mjsraman slash long slash very long is getting removed, ok? Let us see how to do this.

(Refer Slide Time: 03:05)

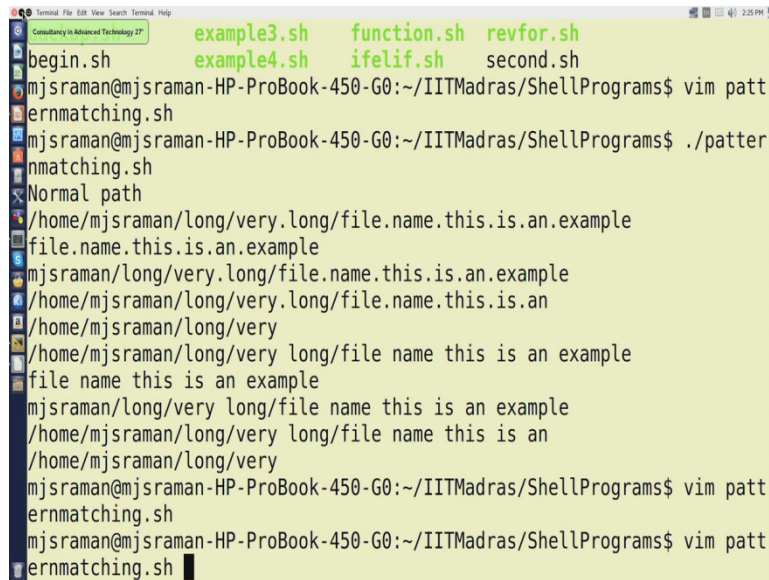


```
me/mjsraman/long/very.long/file.name.this.is.an.example
2
3 echo "Normal path"
4 echo $path
5
6 #Delete the longest path from the beginning that matches
7 echo ${path##/*/}
8
9 #Delete the longest path from the beginning that matches
10 echo ${path#/*/}
11
12 #Delete the shortest path from the end that matches
13 echo ${path%.*}
14
15 #Delete the longest path from the end that matches
16 echo ${path%.*}
17
18
19 IFS=.
20 echo $path
"patternmatching.sh" 24L, 471C
```

So let us go to this pattern matching program and it says that echo path hash hash slash start slash ok so essentially what it means is that delete the longest path from the beginning that matches the following, ok so longest path from the beginning that matches the following so let us go back to this example so what is the longest path from the beginning to match the following.

So essentially if you look at this this slash goes here, ok? And if you look at this this slash with a slash and the longest path within these two is nothing but your slash home slash mjsraman slash long slash very dot long slash filename so essentially what you do is you want to remove match all this ok and then you want to just print file dot name dot this is an example.

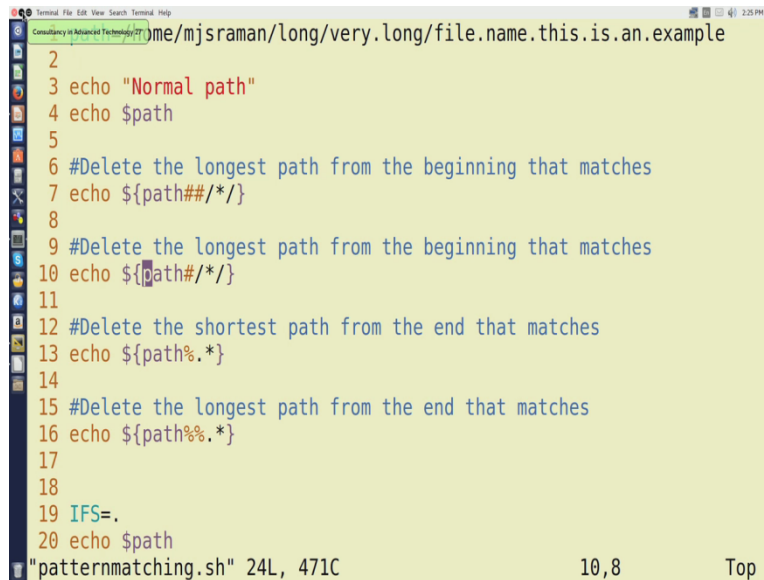
(Refer Slide Time: 04:08)



```
Terminal File Edit View Search Terminal Help
Consolidating in Advanced Technology 27
example3.sh  function.sh  revfor.sh
begin.sh     example4.sh  ifelif.sh   second.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patternmatching.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./patternmatching.sh
Normal path
/home/mjsraman/long/very.long/file.name.this.is.an.example
file.name.this.is.an.example
mjsraman/long/very.long/file.name.this.is.an.example
/home/mjsraman/long/very.long/file.name.this.is.an
/home/mjsraman/long/very
/home/mjsraman/long/very long/file name this is an example
file name this is an example
mjsraman/long/very long/file name this is an example
/home/mjsraman/long/very long/file name this is an
/home/mjsraman/long/very
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patternmatching.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patternmatching.sh
```

So if you want to do that ok let's look at this so you are doing this here so what happens was we gave a hash hash, so if you want you gave a hash hash whatever starts with slash and then ends with the slash and between that whatever is there was the longest one was removed ok and so therefore what it got printed was file dot name dot this. is an example. Now let us take the next one ok so in the next example so the next print that we see is mjsraman which essentially means slash home slash disappeared. So how to remove the first shortest part is a string, ok is given by this command, ok?

(Refer Slide Time: 04:48)



```
me/mjsraman/long/very.long/file.name.this.is.an.example
2
3 echo "Normal path"
4 echo $path
5
6 #Delete the longest path from the beginning that matches
7 echo ${path##*/}
8
9 #Delete the longest path from the beginning that matches
10 echo ${path#*/}
11
12 #Delete the shortest path from the end that matches
13 echo ${path%.*}
14
15 #Delete the longest path from the end that matches
16 echo ${path%.*}
17
18
19 IFS=.
20 echo $path
"patternmatching.sh" 24L, 471C 10,8 Top
```

So if you look at this command I have echo path, path is what this variable carries and then you just take if you put one hash it means I have to remove all those shortest path within two hashes. Now if you look at the two hashes that exist, this is the first hash and this is the sorry, this is the first slash, so this is the first slash and this is the next slash. In between these two whom exist and that is the shortest one that matches between these two, therefore you get home removes and it actually prints mjsraman long and so on the all the rest of the stuff.

Similarly the next variable it says path percentage dot ok star ok so let us so essentially what does it mean delete the shortest part from the end that matches, so guess what does it going to print, if your guess is right it is going to print up to and is this correct, let's see.

(Refer Slide Time: 05:53)

```
Terminal File Edit View Search Terminal Help
/home/mjsraman/long/very
/home/mjsraman/long/very long/file name this is an example
file name this is an example
mjsraman/long/very long/file name this is an example
/home/mjsraman/long/very long/file name this is an
/home/mjsraman/long/very
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patternmatching.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patternmatching.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patternmatching.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./patternmatching.sh
Normal path
/home/mjsraman/long/very.long/file.name.this.is.an.example
file.name.this.is.an.example
mjsraman/long/very.long/file.name.this.is.an.example
/home/mjsraman/long/very.long/file.name.this.is.an
/home/mjsraman/long/very
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$
```

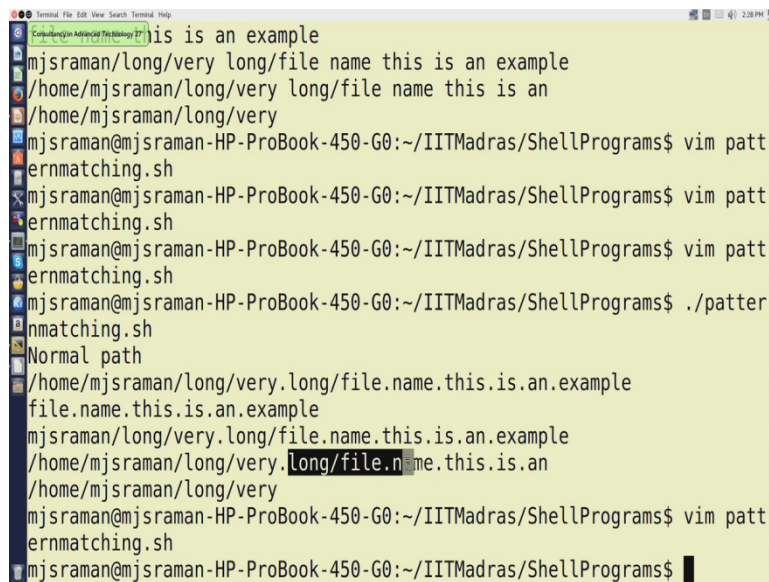
So let us look at the third line that that executes after this, so the third line that executes is we have the normal path then we have the first path and then the third line after the executes is this, you see home mjsraman, ok? Long very long file name also we dropped that example ok which is what the expected ok and finally when we put.

(Refer Slide Time: 06:21)

```
Terminal File Edit View Search Terminal Help
/home/mjsraman/long/very.long/file.name.this.is.an.example
2
3 echo "Normal path"
4 echo $path
5
6 #Delete the longest path from the beginning that matches
7 echo ${path##*/}
8
9 #Delete the longest path from the beginning that matches
10 echo ${path#*/}
11
12 #Delete the shortest path from the end that matches
13 echo ${path%.*}
14
15 #Delete the longest path from the end that matches
16 echo ${path%.*}
17
18
19 #IFS=.
20 #echo $path
:wq
```

When we put echo path percentage percentage dot star which essentially tells you that you should go for a longest match that is there from a dot, so let us take a look at this.

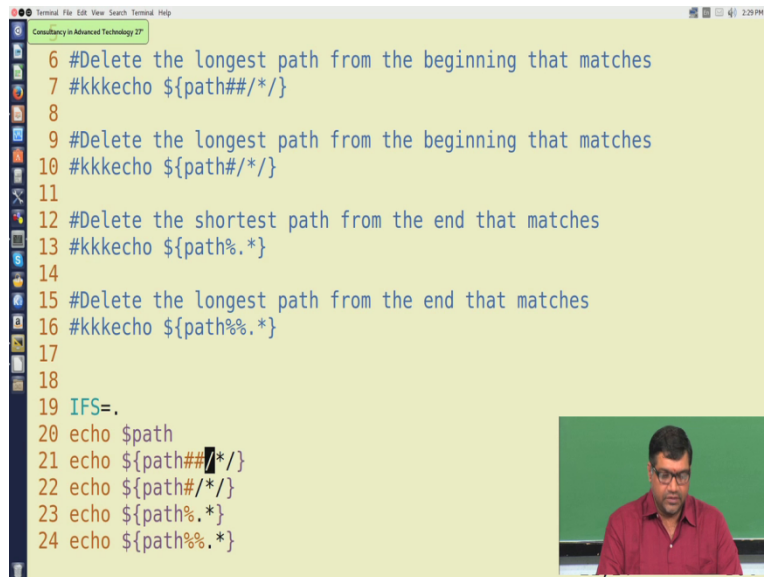
(Refer Slide Time: 06:34)



```
Terminal File Edit View Search Terminal Help
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patternmatching.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./patternmatching.sh
Normal path
/home/mjsraman/long/very.long/file.name.this.is.an.example
file.name.this.is.an.example
mjsraman/long/very.long/file.name.this.is.an.example
/home/mjsraman/long/very.long/file.name.this.is.an
/home/mjsraman/long/very
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patternmatching.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$
```

What is a longest match from a dot so if you look at this this very dot long is a longest match from the end ok because after this there is no dot here ok therefore it removes all these values ok after this dot ok and then it prints only home mjsraman long dot very, I hope this clears you on how this pattern matching can work with respect to dollar I mean this manipulating the path variable now let us now go ahead and then remove.

(Refer Slide Time: 07:15)

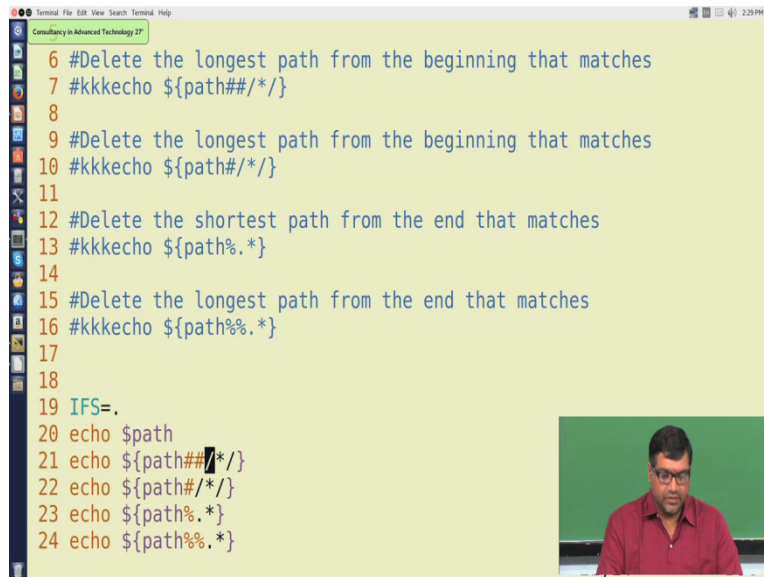


```
6 #Delete the longest path from the beginning that matches
7 #kkkecho ${path##*/}
8
9 #Delete the longest path from the beginning that matches
10 #kkkecho ${path#*/}
11
12 #Delete the shortest path from the end that matches
13 #kkkecho ${path%.*}
14
15 #Delete the longest path from the end that matches
16 #kkkecho ${path%.*}
17
18
19 IFS=.
20 echo $path
21 echo ${path##*/}
22 echo ${path#*/}
23 echo ${path%.*}
24 echo ${path%.*}
```

The commands of the last five lines so we go and command out these first five lines ok and we'll go and and then we'll remove the commands of the last five lines we'll just leave the path example there so let us make so as I told you before ISF is the internal field separator so now what we will do is previously our default ISF was slash and now we'll remove to dot an internal field separator the dot ok now after this let's see how these command will work ok so we have moved the internal field separator dot and let's try to see how we can work with these commands so if the internal field separator is dot then I told you that, it will be reading it as a string

So let's run this program.

(Refer Slide Time: 08:07)



```
6 #Delete the longest path from the beginning that matches
7 #kkkecho ${path##*/}
8
9 #Delete the longest path from the beginning that matches
10 #kkkecho ${path#*/}
11
12 #Delete the shortest path from the end that matches
13 #kkkecho ${path%.*}
14
15 #Delete the longest path from the end that matches
16 #kkkecho ${path%.*}
17
18
19 IFS=.
20 echo $path
21 echo ${path##*/}
22 echo ${path#*/}
23 echo ${path%.*}
24 echo ${path%.*}
```

So in this case the normal path is home mjsraman long very dot long slash file name the and so on, now once I put the internal field separator as dot then what happens is that we are no longer dealing with the single string, we are now dealing with multiple strings.

Take a look at this so if you look at this example so if you look at the path the normal path was slash home mjsraman long very long ok but then look at this the next one, ok? It just the internal field separator being this so the style goes ahead and then removes all those dot and and it just prints the whole string, so what does this do, so what is the longest pattern matching so the longest pattern matching is the same stuff itself ok, and then so what happens here is you print the path and then once you have the internal field separator so let's let's go on take a look at the program, so if you look at this.

(Refer Slide Time: 09:13)

```
Terminal File Edit View Search Terminal Help
Consistency in Advanced Technology 27
6 #Delete the longest path from the beginning that matches
7 #kkkecho ${path##*/}
8
9 #Delete the longest path from the beginning that matches
10 #kkkecho ${path#*/}
11
12 #Delete the shortest path from the end that matches
13 #kkkecho ${path%.*}
14
15 #Delete the longest path from the end that matches
16 #kkkecho ${path%.*}
17
18
19 IFS=.
20 echo $path
21 echo ${path##*/}
22 echo ${path#*/}
23 echo ${path%.*}
24 echo ${path%.*}
20,5 Bot
```

Program so the first time you print the echo path you get this printed so there you have all those slashes and dots, the next time you print echo path you see that because I used ISF to be dot then it prints some bunch of strings ok?

(Refer Slide Time: 09:30)

```
Terminal File Edit View Search Terminal Help
Consistency in Advanced Technology 27
mjan/long/very.long/file.name.this.is.an.example
file.name.this.is.an.example
mjsraman/long/very.long/file.name.this.is.an.example
/home/mjsraman/long/very.long/file.name.this.is.an
/home/mjsraman/long/very
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patt
ernmatching.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patt
ernmatching.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./patter
nmatching.sh
Normal path
/home/mjsraman/long/very.long/file.name.this.is.an.example
/home/mjsraman/long/very long/file name this is an example
file name this is an example
mjsraman/long/very long/file name this is an example
/home/mjsraman/long/very long/file name this is an
/home/mjsraman/long/very
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patt
ernmatching.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$
```

And those strings have a space so if you look at this the next time it prints a path then it removes all the dot and then puts this as an example.

(Refer Slide Time: 09:40)

```
Terminal File Edit View Search Terminal Help
Consistency in Advanced Technology 27

6 #Delete the longest path from the beginning that matches
7 #kkkecho ${path##*/}
8
9 #Delete the longest path from the beginning that matches
10 #kkkecho ${path#*/}
11
12 #Delete the shortest path from the end that matches
13 #kkkecho ${path%.*}
14
15 #Delete the longest path from the end that matches
16 #kkkecho ${path%.*}
17
18
19 IFS=.
20 echo $path
21 echo ${path##*/}
22 echo ${path#*/}
23 echo ${path%.*}
24 echo ${path%.*}
:
```

So now if you go to the third part so it says all the slashes with the longest slash slash tag ok, so now what happens is that.

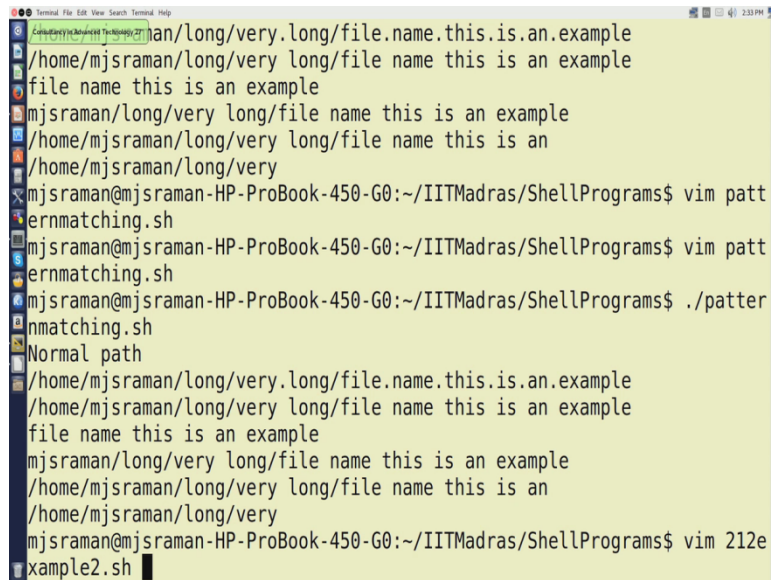
(Refer Slide Time: 09:50)

```
Terminal File Edit View Search Terminal Help
Consistency in Advanced Technology 27

mjsraman@mjsraman-HP-ProBook-450-G0:~/long/very.long/file.name.this.is.an
/home/mjsraman/long/very
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patt
ernmatching.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patt
ernmatching.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./patter
nmatching.sh
Normal path
/home/mjsraman/long/very.long/file.name.this.is.an.example
/home/mjsraman/long/very long/file name this is an example
file name this is an example
mjsraman/long/very long/file name this is an example
/home/mjsraman/long/very long/file name this is an
/home/mjsraman/long/very
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patt
ernmatching.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patt
ernmatching.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./patter
nmatching.sh
```

It's a longest path so if you look at that the example here. Ok we'll run this.

(Refer Slide Time: 09:57)

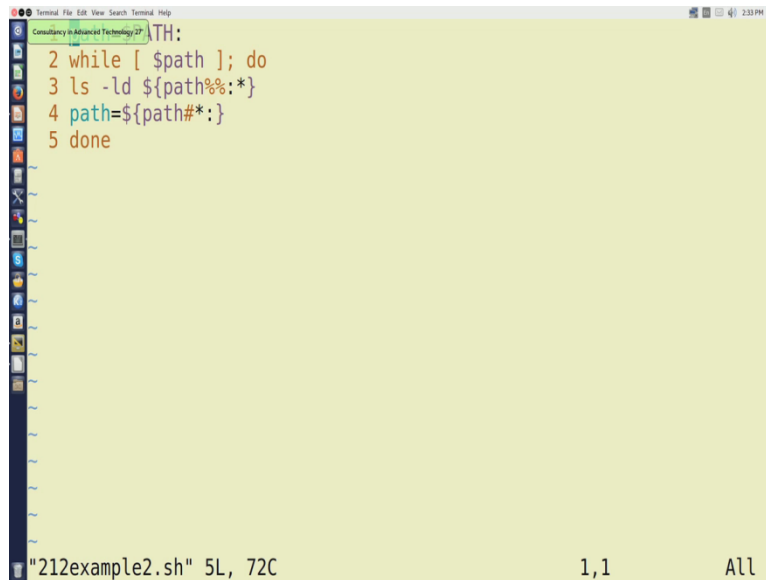


```
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patternmatching.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patternmatching.sh
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./patternmatching.sh
Normal path
/home/mjsraman/long/very.long/file.name.this.is.an.example
/home/mjsraman/long/very long/file name this is an example
file name this is an example
mjsraman/long/very long/file name this is an example
/home/mjsraman/long/very long/file name this is an
/home/mjsraman/long/very
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim 212example2.sh
```

So look at this the longest path ok? The longest path is if you look at this it starts from home mjsraman long very long and after that the file name this is an example is a longest path, then it takes a shortest path. The shortest path is you remove home then everything else that is printed as a string and then finally from the reverse you go up to example example goes out and then you go up to long goes out ok.

So based on this pattern matching let us try to see a very small example and let us see how powerful this pattern matching is. So let's consider this example of using pattern matching it's a very small example but then it's very powerful ok?

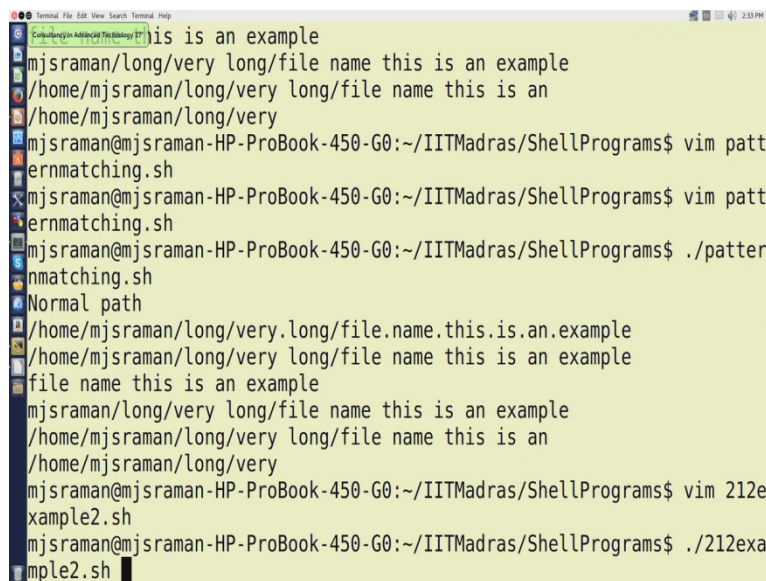
(Refer Slide Time: 10:42)



```
Terminal File Edit View Search Terminal Help
Consulency in Advanced Technology 27
mjsr@TH:
2 while [ $path ]; do
3 ls -ld ${path%:*}
4 path=${path#*:}
5 done
"212example2.sh" 5L, 72C 1,1 All
```

So let us take you, so essentially what this does is we'll first run the program and then try to understand how this pattern matching works ok?

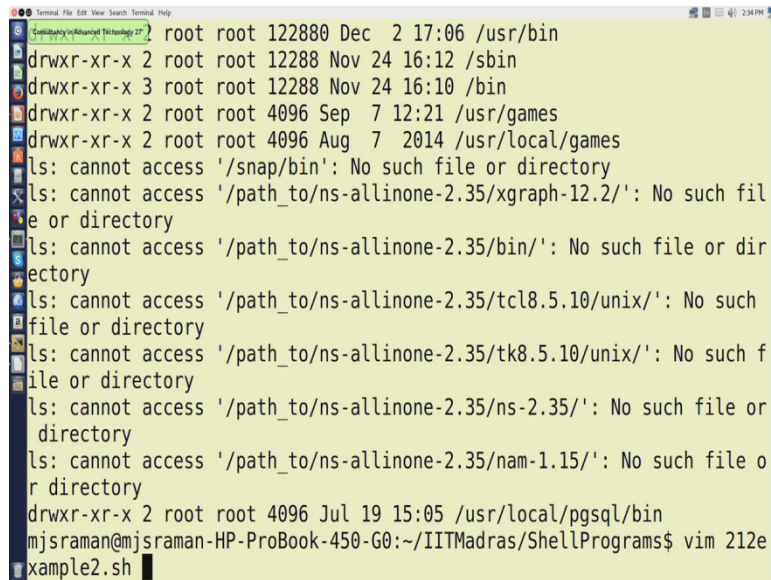
(Refer Slide Time: 10:54)



```
Terminal File Edit View Search Terminal Help
Consulency in Advanced Technology 27
mjsr@this is an example
/home/mjsraman/long/very long/file name this is an example
/home/mjsraman/long/very long/file name this is an
/home/mjsraman/long/very
mjsr@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patt
ernmatching.sh
mjsr@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim patt
ernmatching.sh
mjsr@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./patter
nmatching.sh
Normal path
/home/mjsraman/long/very.long/file.name.this.is.an.example
/home/mjsraman/long/very long/file name this is an example
file name this is an example
mjsr@mjsraman/long/very long/file name this is an example
/home/mjsraman/long/very long/file name this is an
/home/mjsraman/long/very
mjsr@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim 212e
xample2.sh
mjsr@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ ./212exa
mple2.sh
```

So let's run this program.

(Refer Slide Time: 11:00)

A terminal window showing the output of a command. The output lists several directories with their permissions, owner, group, size, date, and path. The paths listed are /usr/bin, /sbin, /bin, /usr/games, /usr/local/games, /snap/bin, /path_to/ns-allinone-2.35/xgraph-12.2/, /path_to/ns-allinone-2.35/bin/, /path_to/ns-allinone-2.35/tcl8.5.10/unix/, /path_to/ns-allinone-2.35/tk8.5.10/unix/, /path_to/ns-allinone-2.35/ns-2.35/, /path_to/ns-allinone-2.35/nam-1.15/, and /usr/local/pgsql/bin. The terminal shows several 'ls: cannot access' error messages for the paths that do not exist. The prompt at the bottom is 'mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms\$ vim 212example2.sh'.

```
root root 122880 Dec  2 17:06 /usr/bin
drwxr-xr-x 2 root root 12288 Nov 24 16:12 /sbin
drwxr-xr-x 3 root root 12288 Nov 24 16:10 /bin
drwxr-xr-x 2 root root 4096 Sep  7 12:21 /usr/games
drwxr-xr-x 2 root root 4096 Aug  7 2014 /usr/local/games
ls: cannot access '/snap/bin': No such file or directory
ls: cannot access '/path_to/ns-allinone-2.35/xgraph-12.2/': No such file or directory
ls: cannot access '/path_to/ns-allinone-2.35/bin/': No such file or directory
ls: cannot access '/path_to/ns-allinone-2.35/tcl8.5.10/unix/': No such file or directory
ls: cannot access '/path_to/ns-allinone-2.35/tk8.5.10/unix/': No such file or directory
ls: cannot access '/path_to/ns-allinone-2.35/ns-2.35/': No such file or directory
ls: cannot access '/path_to/ns-allinone-2.35/nam-1.15/': No such file or directory
drwxr-xr-x 2 root root 4096 Jul 19 15:05 /usr/local/pgsql/bin
mjsraman@mjsraman-HP-ProBook-450-G0:~/IITMadras/ShellPrograms$ vim 212example2.sh
```

So essentially what this program does is ok it tries to identify whatever we had done previously it does tries to do the same thing ok. So if you remember previously we had tried to do the same thing we achieved it in a different method, so this program we achieved it using the for loop so in this case what we are trying to do is we are trying to achieve the same result in a different way.

(Refer Slide Time: 11:27)

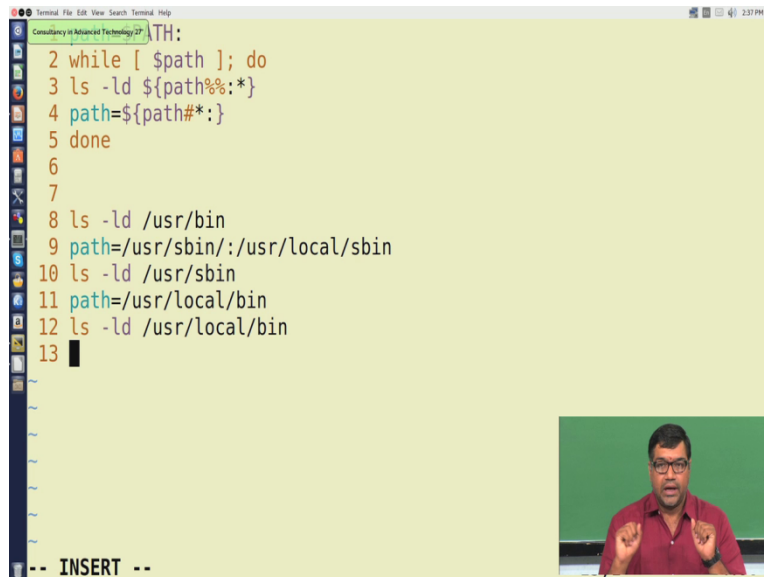


```
Terminal: File Edit View Search Terminal Help
Consistency in Advanced Technology 27
ATH:
2 while [ $path ]; do
3 ls -ld ${path%:*}
4 path=${path#*:}
5 done
6
7
8 /usr/bin
9 /usr/sbin:/usr/local/sbin
```

-- INSERT --

So using string matching so how does this work? So if you remember I had told you that with string matching ok you could match parts of a string and then go on working it in a loop ok in the previous example in we actually worked with with respect to a for loop so we took a for loop and found out the internal field separator of the for loop and then based on the internal field separator what we did was we went ahead and then put back in the for loop. Now what we are trying to do is we are taking the whole path variable so this while dollar path implies if the path variable becomes null ok then exit the loop if the path variable does become null, ok? What you we do is you go ahead and list out the directory, now how do you list out the directory you take the, now remember this in the previous example we saw percent percent colon star, what does it now this colon was the field separator so what does that mean you try to take the longest path ok? And then you had to remove that longest path take the first few paths I mean from the end, you remove the longest path from the end, take the first path and then print directory so let us say for example I had something like this so let's say I had example slash user slash bin so let us say I had something like this slash user slash bin colon and I had slash user slash sbin and then I had something like slash user slash local slash sbin, ok? Let us say I have something like this, Now the third line what it essentially says is that it say it says ok split the string something like this ok hope you understand.

(Refer Slide Time: 13:29)



```
TH:
2 while [ $path ]; do
3 ls -ld ${path%:*}
4 path=${path#*:}
5 done
6
7
8 ls -ld /usr/bin
9 path=/usr/sbin:/usr/local/sbin
10 ls -ld /usr/sbin
11 path=/usr/local/bin
12 ls -ld /usr/local/bin
13
```

So it says remove the longest match from the end and once I removed the longest match then what is the result you see that this will be executed as ls minus ld minus ld. So this will be executed something like this ok which is actually what we want and then what happens in the next line so if in the next line what happens is the path variable is again initialized to what, the first component of the shortest of the shortest match.

So the first common, so the path variable becomes something like this the path variable in the next line become something like this, again it goes into your loop since the path variable is not null, what happens next time is let us say it becomes like this slash user slash sbin ok and then the path is initialized too. Now user local bin slash local slash bin and then finally I print minus ld slash user slash local slash bin so this is how the program executes so if you look at this this the variation of printing the correct the existing directories in the path variable ok partial variable and this is now this is another way of writing the program. In the previous program actually you saw that we had actually used the for loop and then split the string so once we entered in the for loop and then we gave the IFSO if you remember in the previous example that we saw we put dot as the IFS and once you put dot as IFS then the string of split based on the dot.

So similarly ok in the for loop we put the IFS as colon and therefore the whole long string of the path variable was split according to the colon as the field separator. Now the point is that this the

both of these program achieve the same end result but except that they achieved it in a different way. Now which one is efficient and all that is the question of debate I mean hopefully this this method is much more efficient, hopefully we I mean we we can just show that this method is much more efficient and but it's it's slightly difficult to understand so this pattern matching ok is something very important, you need to ensure that you you learn about this in shell scripting, it can make your life extremely easy when you do these kind of scripts.